A Fast and Space-Efficient Boundary Element Method for Computing Electrostatic and Hydration Effects in Large Molecules

RANDY J. ZAUHAR*

Tripos, Inc., 1699 S. Hanley Road, St. Louis, Missouri 63144

ALEXANDER VARNEK

Department of Chemistry, University of Strasbourg, 4 Rue Blaise Pascal, Strasbourg, France

Received 21 February 1995; accepted 5 July 1995

ABSTRACT

At present, there are two widely used approaches for computing molecular hydration and electrostatic effects within the continuum approximation: the finite difference method, in which the electric potential is directly computed on a cubic grid, and the induced polarization charge or boundary element method, in which an induced charge distribution is first computed on the molecular surface and in which solvation effects are then calculated by reference to the reaction field arising from this induced surface charge. While the induced surface charge approach has a number of advantages over finite differences, especially in the computation of hydration forces and solvent stabilization, the applications of this technique have been largely restricted to small molecules. This is primarily due to the very large system of equations that results when the surface of a macromolecule is discretized into elements small enough to ensure an acceptable level of numerical accuracy within the continuum model. This article describes a new algorithm for implementing boundary element calculations within the continuum model. The essence of our approach is only to compute explicitly those interactions between surface elements that are relatively close together and to approximate long-range interactions by grid-based multipole expansion. The resulting system of equations has a relatively sparse coefficient matrix and requires disk storage that increases linearly with molecular surface area. The technique has numerous applications in the analysis of solvation effects in large molecules, especially in the area of conformational analysis, where it is critical to accurately estimate the global hydration energy for the entire structure.

*Author to whom all correspondence should be addressed.

Journal of Computational Chemistry, Vol. 17, No. 7, 864–877 (1996)

CCC 0192-8651 / 96 / 070864-14

Introduction

Continuum solvation models are employed to compute hydration effects in many studies of biologically important molecules. Applications include estimation of pK shifts in proteins,[1-3] calculation of solvent effects in molecular orbital calculations,[4-7] molecular mechanics and dynamics studies of small molecules,[8-10] and analysis of electrostatic effects in binding and subunit assembly.[11-13] With the exception of the Tanford-Kirkwood model and its descendants,[14,15] continuum methods treat a solvated molecule as a cavity of arbitrary shape embedded in a continuous solvent of high dielectric constant. The interior of the cavity is assigned a low dielectric constant (appropriate for the molecular interior) and contains the solute charge distribution. (Some formulations of the problem admit more structure in the interior and include representations of atomic polarizability.[16]) Continuum methods proceed by solving either the Poisson equation[17-19] or the Poisson-Boltzmann equation.[20,21] The electrostatic portion of hydration is then represented by that component of the total electric potential that arises from the polarization of the solvent.

The cavity defined by the molecular surface has a complicated geometry, and solution of the electrostatic problem in this situation requires a numerical technique. The two approaches in wide use are the finite difference method (FDM), in which the electric potential is solved directly on a three-dimensional grid that encloses the solute and a portion of the surrounding solvent, and the induced surface charge or boundary element method (BEM), in which a distribution of induced charge (arising from the polarization of the solvent) is computed over the molecular surface. In the finite difference approach,[1-3,8,11,13,17,20] the total potential is immediately available at any point by interpolation from neighboring grid vertices, while in the BEM[4-7,9,10,12,16,18,19,21] the determination of the total potential requires an auxiliary calculation involving integration over the distribution of induced surface charge. On the other hand, in the FDM it is necessary to use special techniques to extract the hydration energies[11] and forces[22-24] from the total electric potential, while in the BEM these quantities are readily computed.

To date, the majority of continuum calculations for macromolecules have involved the FDM, while the BEM has been applied primarily to small molecules (often in the context of molecular orbital computations). This is despite the fact that the BEM would appear to have an advantage over finite differences for large systems, since the molecular surface (the region discretized in the BEM) increases in size much more slowly than the molecular volume (which, along with surrounding solvent, is discretized in the FDM). This situation is partly historical, since the first continuum method for macromolecules used finite differences,[17] while the boundary element approach was first applied to small molecules.[18]

However, one must also consider the kinds of studies that have been carried out with these methods. Finite difference computations usually employ a fixed grid spacing, and while "focusing"[25] has been used to maintain numerical accuracy in localized regions, no attempt is made to ensure precision over entire structures. In fact, it has been shown that a rather fine grid is necessary to achieve convergent energies for small molecules,[26] and it would clearly be impractical to apply this level of precision to a large system using the FDM. In contrast, BEM applications often involve attempts to estimate global quantities, such as the total electrostatic component of the hydration free energy, or the forces of hydration acting over an entire molecule. Partitioning the surface of a macromolecule into elements as small as those typically used in BEM computations for small molecules leads to a large number of surface elements, and consequently a very large system of simultaneous equations to solve in order to determine the distribution of induced surface charge. (Indeed, the relatively few BEM calculations for large molecules have made no attempt at maintaining global accuracy, and in this sense they have suffered from the same computational limitation as the FDM.)

Applying either the FDM or the BEM to a macromolecule with the level of discretization required to maintain an acceptable level of numerical accuracy can lead to a system of equations of unwieldy size. For example, to cover the surface of a small protein with elements of approximately 1 Å on a side leads to a triangulation that will include about 20,000 elements, or 10,000 nodes (element corners). The electrostatic problem will then be defined by 10,000 simultaneous equations; the resulting coefficient matrix will require 400 megabytes of storage in single (4-byte) precision. Larger molecules (or finer discretization of the

surface) will lead to rapidly increasing demands for temporary storage and also for central processing unit (CPU) time when solving the system of equations. The potentially massive demands on computing resources are without doubt an important reason that the BEM has not been extensively applied to large molecules and that computations involving proteins have usually involved a rather coarse discretization of the surface.[12,21]

This article presents a new boundary element method for computing electrostatic and hydration effects in both small and large biomolecules. The method, which uses a truncated approximation of the full coefficient matrix, is designed to allow optimal control over the utilization of computing resources. It is shown that the technique can provide solutions that are nearly identical to those produced using a complete coefficient matrix, but with much reduced requirements for computing resources (especially storage). The method is demonstrated with the ion-binding molecule calixarene and with the proteins lysozyme and calmodulin.

Methods

A general formalism for solving the Poisson equation in the context of molecular electrostatics has been presented previously.[19,27,28] In brief, the solvent accessible molecular surface is discretized into a collection of curvilinear three-sided finite elements. Each element has three corners, or nodes, and there is assumed to exist a continuous, differentiable mapping from a standard unit triangle in the r-s plane to each element. The induced polarization charge density $\sigma^m(r, s)$ within element m is related by linear interpolation to the values at the three element corners $[m(1), m(2), m(3)]$:

$$\sigma^m(r, s) = (1 - r - s)\sigma_{m(1)} + r\sigma_{m(2)} + s\sigma_{m(3)} \quad (1)$$

Each element corner (node) i has associated surface normal $\mathbf{n}_i$ and total normal electric field E_i. The normal field can be expressed as the sum of two components, arising from either the internal charge distribution of the solute $[E^{(\mathrm{int})}i]$ or from the distribution of induced surface charge $[E^{(\mathrm{srf})}i]$. Explicitly,

$$E^{(\mathrm{int})}{}_i = \frac{1}{D_i}\sum_j q_j \frac{(\mathbf{r}_i - \mathbf{r}_j)\cdot \mathbf{n}_i}{|\mathbf{r}_i - \mathbf{r}_j|^3} \quad (2)$$

where the index j runs over solute charges $\{q_j\}$ with positions $\{\mathbf{r}_j\}$, and D_i is the dielectric constant of the solute. Furthermore,

$$E^{(\mathrm{srf})}{}_i = \sum_t K_{it}\sigma_t \quad (3)$$

where the index t runs over all the nodes, and coefficients K_{it} are found by integrating over the molecular surface. Each surface element j gives rise to a contribution to the normal field at node i given by

$$E_i^{(j)} = K_{i,j(1)}^{(j)}\sigma_{j(1)} + K_{i,j(2)}^{(j)}\sigma_{j(2)} + K_{i,j(3)}^{(j)}\sigma_{j(3)} \quad (4)$$

where

$$K_{i,j(1)}^{(j)} = \int_j (1 - r - s)\frac{(\mathbf{r}_i - \mathbf{r}(r, s))\cdot \mathbf{n}_i}{|\mathbf{r}_i - \mathbf{r}(r, s)|^3} \times \det[J(r, s)]\, dr\, ds$$

$$K_{i,j(2)}^{(j)} = \int_j r\frac{(\mathbf{r}_i - \mathbf{r}(r, s))\cdot \mathbf{n}_i}{|\mathbf{r}_i - \mathbf{r}(r, s)|^3}\det[J(r, s)]\, dr\, ds \quad (5)$$

and

$$K_{i,j(3)}^{(j)} = \int_j s\frac{(\mathbf{r}_i - \mathbf{r}(r, s))\cdot \mathbf{n}_i}{|\mathbf{r}_i - \mathbf{r}(r, s)|^3}\det[J(r, s)]\, dr\, ds$$

Here $J(r, s)$ is the Jacobian of the mapping from the unit triangle to the element j. The coefficients K_{it} in eq. (3) are then given by

$$K_{it} = \sum_{(j,\varepsilon)} K_{i,j(\varepsilon)}^{(j)} \qquad (1 \le \varepsilon \le 3) \quad (6)$$

where the summation is taken over all pairs (j, ε) with $j(\varepsilon) = t$. This process of equation assembly is carried out for each row (node) i, producing a square coefficient matrix $\mathbf{K}$ with dimension equal to the number of vertices defined by the triangulation of the surface. The vector $[\sigma]$ of nodal polarization densities is then found as a solution of the system of equations

$$\{\mathbf{I} - f\mathbf{K}\}[\sigma] = f\mathbf{E}^{(\mathrm{int})} \quad (7)$$

with $\mathbf{I}$ the unit matrix, $\mathbf{E}^{(\mathrm{int})}$ the vector of the normal electric field at the nodes due to the solute charge distribution, and f a constant that depends only on the dielectric constants assumed for the solvent and the solute interior.[19]

The goal of our development is not to compute the entire matrix **K**, but rather to compute a truncated version in which most of the coefficients are identically zero. This is accomplished by a grid multipole method, analogous to (although less complex than) those that have been developed for computing nonbonded interactions in molecular mechanics.[29]

We begin by enclosing the molecular surface in a rectangular grid with spacing Δ. The grid has N_x, N_y, and N_z divisions along the x, y, and z axes, respectively, and is just large enough to enclose all the nodes. The grid divides the enclosed volume into $N_x N_y N_z$ cubical elements. As illustrated in Figure 1, each cubical element has grid coordinates (i, j, k) and midpoint position $\mathbf{p}(i, j, k)$. Each node falls inside a particular volume element, and so too each surface element m is associated with exactly one volume element, or cube, $c(m)$, with grid coordinates $ic(m)$, $jc(m)$, and $kc(m)$. If the nodes at the corners of element m all fall inside different cubes, then the element is associated with the cube that contains the first node of the element; if any cube contains two or more of the nodes, then the element is associated with that cube. In addition, for each cube we maintain a list of elements that are thus "attached" to it. For a typical molecular surface and grid, most of the cubical elements will be empty (associated with no surface elements).

We assume that some estimate of the polarization charge distribution is always available (for simplicity, the initial distribution can be taken as zero). For each element, we compute a collection of coefficients that allow the monopole, dipole, and quadrupole moments of the element with respect to the associated cube center to be rapidly computed, given the polarization charge densities at the element corners. For the monopole moment of element m, we find

$$q^m_{c(m)} = q^m_{c,1}\sigma_{m(1)} + q^m_{c,2}\sigma_{m(2)} + q^m_{c,3}\sigma_{m(3)} \qquad (8)$$

where $c(m)$ is the cube associated with the element. The first coefficient is easily seen to be

$$q^m_{c,1} = \int_m (1 - r - s)\det[J(r, s)]\, dr\, ds \qquad (9)$$

The dipole moment vector relative to the cube center can be written

$$\mathbf{d}^m_{c(m)} = \mathbf{d}^m_{c,1}\sigma_{m(1)} + \mathbf{d}^m_{c,2}\sigma_{m(2)} + \mathbf{d}^m_{c,3}\sigma_{m(3)} \qquad (10)$$

and the first vector coefficient is given by

$$\mathbf{d}^m_{c,1} = \int_m (1 - r - s)(\mathbf{r}(r, s) - \mathbf{p}(c)) \times \det[J(r, s)]\, dr\, ds \qquad (11)$$

where $\mathbf{r}(r, s)$ locates a point on the element with local coordinates (r, s), and $\mathbf{p}(c)$ is the midpoint of the cube that the element belongs to.

The quadrupole moment of the element m with respect to $\mathbf{p}$ is expressed as a matrix of nine linear operators acting on the space of node polarization densities:

$$[\mathbf{u}^m_c] = \begin{bmatrix} u^m_{c,11} & u^m_{c,12} & u^m_{c,13} \\ u^m_{c,21} & u^m_{c,22} & u^m_{c,23} \\ u^m_{c,31} & u^m_{c,32} & u^m_{c,33} \end{bmatrix} \qquad (12)$$

The term $u^m_{c,ij}$ is given by

$$u^m_{c,ij} = u^m_{c,ij,1}\sigma_{m(1)} + u^m_{c,ij,2}\sigma_{m(2)} + u^m_{c,ij,3}\sigma_{m(3)} \qquad (13)$$

where, for example

$$u^m_{c,ij,1} = \int_m (1 - r - s)\big[3(x_i(r, s) - p_i) \times (x_j(r, s) - p_j) - |\mathbf{r}(r, s) - \mathbf{p}|^2\delta_{ij}\big] \otimes \det[J(r, s)]\, dr\, ds \qquad (14)$$

with $\mathbf{p} = \{p_1, p_2, p_3\}$ the cube midpoint, $\mathbf{r}(r, s) = \{x_1(r, s), x_2(r, s), x_3(r, s)\}$ and δ_{ij} the Kronecker delta. Equations (13) and (14) are a straightforward statement of the definition of the quadrupole moment, under the condition of linear interpolation of the polarization density within an element.

The total multipole moments for cube c are found by simply summing the contributions of all the elements that are attached to the cube. If these elements form the set $M(c)$, then the cube monopole, dipole, and quadrupole moments are written

$$q^c = \sum_{m \in M(c)} q^m_c, \qquad \mathbf{d}^c = \sum_{m \in M(c)} d^m_c, \qquad \text{and} \qquad [\mathbf{u}^c] = \sum_{m \in M(c)} [\mathbf{u}^m_c] \qquad (15)$$

The electric field at any point $\mathbf{t}$ due to the multipole moments of cube c is then given by

$$\mathbf{E}^c(\mathbf{t}) = \mathbf{E}^c_q(\mathbf{t}) + \mathbf{E}^c_d(\mathbf{t}) + \mathbf{E}^c_u(\mathbf{t}) \qquad (16)$$

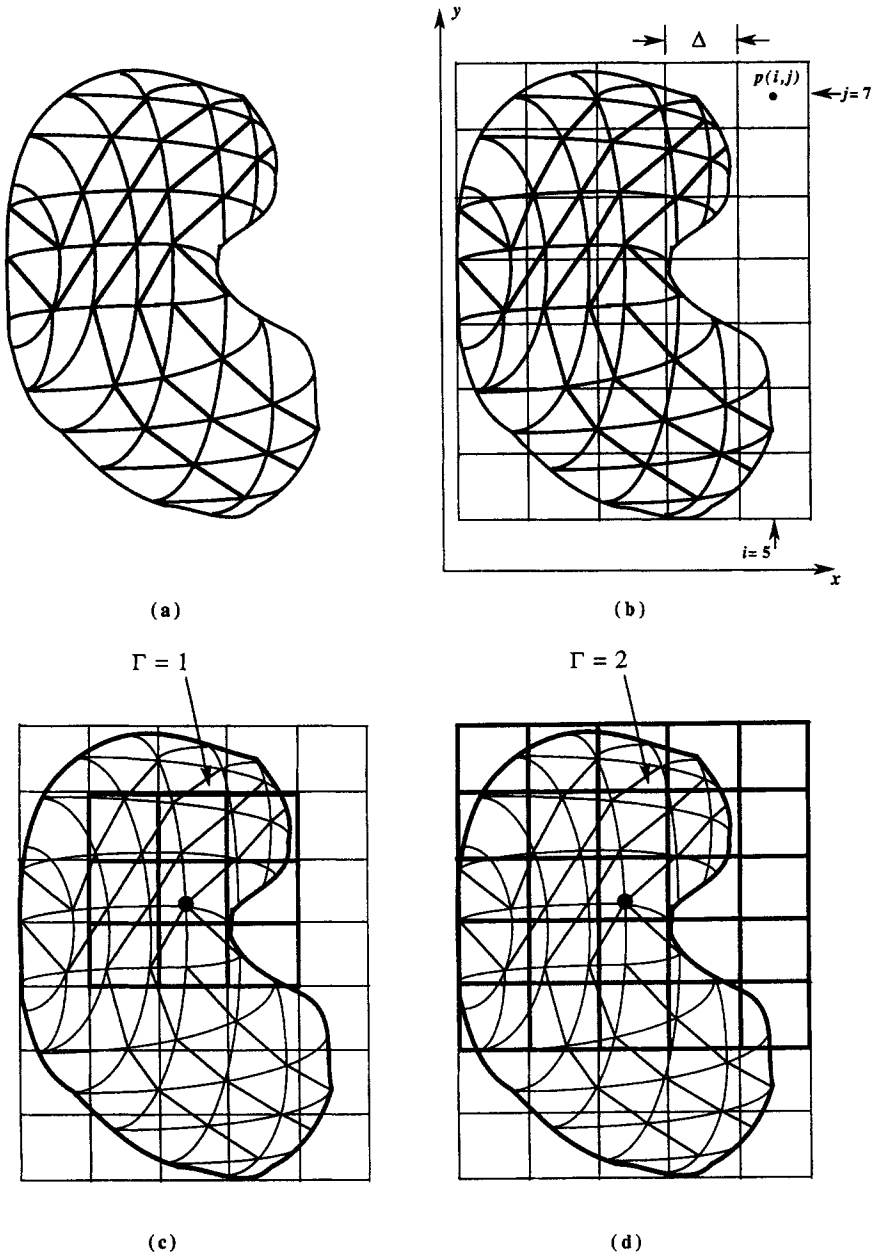


FIGURE 1. (a) Cartoon of a molecule with triangulated solvent accessible surface. (b) A grid with spacing Δ superimposed over the molecule. The grid extends just far enough in each dimension to completely enclose the molecule. Surface elements are assigned to the cubical elements of the grid, as described in the text. Only two dimensions are shown here—the grid is actually three dimensional. (c) Cubes within grid increment $\Gamma = 1$ of the indicated node are highlighted. Explicit integration will be carried out only for the elements that lie within these cubes when constructing the row of the coefficient matrix that corresponds to the indicated node. (d) Same as (c), but with grid increment $\Gamma = 2$.

with the monopole contribution

$$\mathbf{E}_q^c(\mathbf{t}) = q^c \frac{(\mathbf{t} - \mathbf{p}(c))}{|\mathbf{t} - \mathbf{p}(c)|^3} \tag{17}$$

dipole contribution

$$\mathbf{E}_d^c(\mathbf{t}) = \frac{3\mathbf{l}(\mathbf{d}^c \cdot \mathbf{l}) - \mathbf{d}^c}{|\mathbf{t} - \mathbf{p}(c)|^3} \tag{18}$$

(with **l** the unit vector from **p** to **t**), and quadrupole field components

$$E^c_{u,k} = \frac{1}{2}\sum_{i,j=1,3}[\mathbf{u}^c]_{ij}\left\{5\frac{x_i x_j x_k}{|\mathbf{t}-\mathbf{p}(c)|^7} - \frac{(x_i\delta_{jk}+x_j\delta_{ik})}{|\mathbf{t}-\mathbf{p}(c)|^5}\right\} \quad k=1,2,3 \quad (19)$$

where $\mathbf{t} - \mathbf{p}(c) = \{x_1, x_2, x_3\}$.

The total normal electric field due to the surface charge is given by eq. (3); we now split this field contribution into two parts, using the assignment of surface elements to cubes. As shown in Figure 1, two quantities are specified when superimposing the grid on the molecular surface—the grid spacing, already discussed, and the grid increment, Γ, which is used to control the computation of coefficients in the matrix **K**. Given any two cubes c and c', we define the distance between them with the metric

$$d(c,c') = \max\{|ic-ic'|, |jc-jc'|, |kc-kc'|\} \quad (20)$$

where ic, jc, and kc are the three grid coordinates for cube c.

Now, for each node i with associated cube $c(i)$, we can express all of the surface elements as the union of two sets: a "near" set $N(i)$ and a "far" set $F(i)$, as follows:

$$\begin{aligned} m \in N(i) \quad &\text{if} \quad d(c(i),c(m)) \le \Gamma \\ m \in F(i) \quad &\text{if} \quad d(c(i),c(m)) > \Gamma \end{aligned} \quad (21)$$

Given this partition, the coefficient matrix **K** can also be decomposed into two parts:

$$\mathbf{K} = \mathbf{K}_N + \mathbf{K}_F \quad (22)$$

Here the matrix $\mathbf{K}_N$ is formed by assembling contributions of all elements in the near set $N(i)$, while $\mathbf{K}_F$ includes all contributions from elements in the far set $F(i)$.

We can now substitute eq. (22) into eq. (7) and rearrange:

$$\{\mathbf{I} - f\mathbf{K}_N\}[\sigma] = f\{\mathbf{E}^{(\text{int})} + \mathbf{K}_F[\sigma]\} \quad (23)$$

The final step in this development is to approximate the second term in braces on the right-hand side of eq. (23) with an approximation based on the grid multipole expansion. Let $C_F(i)$ be the set of "far" cubes relative to the cube associated with node i. We form the vector **X** with elements given by

$$X_i = \sum_{c\in C_F(i)}\left\{\mathbf{E}^c_q(\mathbf{r}_i)(+\mathbf{E}^c_d(\mathbf{r}_i)(+\mathbf{E}^c_u(\mathbf{r}_i)))\right\}\cdot\mathbf{n}_i \quad (24)$$

where the extra parentheses are used to indicate that the approximation may in fact be truncated at the monopole or dipole level. We now introduce the approximation

$$\mathbf{K}_F[\sigma] \cong \mathbf{X} \quad (25)$$

Finally, we use the multipole approximation [eq. (25)] to redefine the electrostatic problem as a system of simultaneous equations with a relatively sparse coefficient matrix:

$$\{\mathbf{I} - f\mathbf{K}_N\}[\sigma] = f\{\mathbf{E}^{(\text{int})} + \mathbf{X}([\sigma])\} \quad (26)$$

Depending on the choice of grid spacing and grid increment, the number of nonzero coefficients in the system of eq. (26) may be a small fraction of those in the "complete" matrix **K**, with much reduced storage requirements. The disadvantage of this approach is the introduction of an additional computational step in forming the vector **X**. In eq. (26), short-range interactions are computed in detail (coefficients in $\mathbf{K}_N$ are calculated by explicit integration), while long-range interactions are approximated by the cube-centered multipole expansions.

Equation 26 represents a nonlinear system of simultaneous equations; as such it must be solved using an iterative method. A brief description of the algorithm we employ follows:

1. Given the surface triangulation and grid parameters, assign the surface elements to cubes in the grid. Compute the matrix $\mathbf{K}_N$ in accordance with the assembly equations [eqs. (3)–(6)] and the partition of the surface elements into "near" and "far" sets ($N(i)$ and $F(i)$) for each node i, using the metric d [eqs. (20) and (21)]. The coefficients are stored in a compact binary format.
2. Compute coefficients to relate polarization densities at element nodes to the element multipole expansions about the associated cube centers [eqs. (8)–(14)]. The expansions may be truncated at the dipole or monopole level, if desired.
3. Generate an initial estimate $[\sigma]$ of the polarization charge distribution (here it is simply set to zero); compute the normal field component at the each node due to the solute

charges via eq. (2); initialize the iteration count $I = 1$.

Main Loop:

4. If $I = 1$ or $\mathrm{mod}(I, 5) = 0$, calculate element multipole expansions using the coefficients found in step 2 (i.e., calculate the expansions at the beginning of the procedure, and update them after every five steps); find the total multipole moments for each cube by summing the moments of all elements attached to the cube. Using the cube multipole moments, calculate the vector **X** [eq. (24)]. Compute the right-hand side of the system of eq. (26).
5. Carry out one iteration of the Gauss-Seidel algorithm for the system of eq. (26). The result is a new estimate $[\sigma']$ of the polarization densities at the nodes.
6. Compute the weighted percent difference $W\%$ between the solutions $[\sigma]$ and $[\sigma']$:

$$W\% = \sum_i \frac{|\sigma_i - \sigma_i'|}{|\sigma_i|} \times 100\% \qquad (27)$$

If $W\% < W_{\max}$ (a user-selected parameter), then let $[\sigma] = [\sigma']$ and exit the procedure.

7. Let $[\sigma] = [\sigma']$ and increment the iteration count I; if $I > I_{\max}$ (a user-selected parameter), exit the procedure; else, go to step 4 and continue.

The new algorithm, which we call the multipole boundary element method (mBEM), permits a great deal of flexibility in controlling the use of computing resources. In particular, it is possible to increase the accuracy of the computation (in the sense of more nearly reproducing the "full matrix" solution) by either increasing the grid increment and/or grid spacing, at the expense of storage space, or by using a higher multipole expansion, at the expense of computing time. Also, by simultaneously decreasing the grid spacing and increasing the grid increment, it is possible to maintain an approximately constant level of storage while making the grid progressively finer and thus increasing the accuracy of the multipole approximation of the long-range electrostatic interactions.

Application

In the implementation of the algorithm described here, the molecular surface is generated by SMART (*SM*ooth molecul*AR* surface *T*riangulator), a program described elsewhere.[30] SMART produces a solvent accessible surface similar to those defined by Richards[31] and Connolly,[32,33] but with modifications that preclude the presence of self-intersecting surface or cusplike structures. The resulting surface is smooth, with a continuous normal. SMART requires as input a molecule data file, with coordinates and an atom type index for each atom, along with a file of radii for the atom types. The program also takes as input an angle parameter which controls the density of nodes in the surface triangulation.

The mBEM algorithm is implemented as a C program. It requires from the user the following input: molecule name (which locates both an atomic coordinate file and a surface file), the grid spacing (in Å) and grid increment, the multipole expansion level (monopole, dipole, or quadrupole), the maximum number of Gauss-Seidel iterations, a convergence parameter (maximum weighted percent difference between successive iterations), and the dielectric constants assumed for the solute interior and the solvent. The program currently runs on Silicon Graphics, Sun, and IBM RISC workstations.

Here, the method is applied to three molecules: calixarene (*t*-butyl-calix[4]arenetetramide), a small metal-binding molecule,[10] and the proteins lysozyme[34] and calmodulin.[35] (The calixarene molecule is in the "converging" conformation described in ref. 10.) The triangulations of the molecular surfaces of the three molecules are shown in Figure 2. The angle parameter for surface triangulation was set to 60° in each case, leading to a node density of approximately $1/\text{Å}^2$. While this density is a bit smaller than expected in accurate calculations,[9,28] it leads to systems of equations for all the molecules that are small enough to permit full-matrix calculations, and thus to test the accuracy of the multipole expansion method. (Even at this low node density, some of the comparison full-matrix computations required over 200 megabytes of disk storage.) Calculations using the mBEM were carried out for grid spacings (Δ) of 3 Å and 5 Å, with grid increments (Γ) of 1 and 2 for $\Delta = 3$ Å and a single grid increment of 1 when $\Delta = 5$ Å. The three combinations [(3, 1), (3, 2) and (5, 1)] lead to boxes defining the "near" elements about each node of dimension 9, 15, and 15 Å, respectively. For each (Δ, Γ) combination, three separate electrostatic calculations were performed, corresponding to truncation of the cube multipole expansions at the monopole, dipole, or quadrupole level.

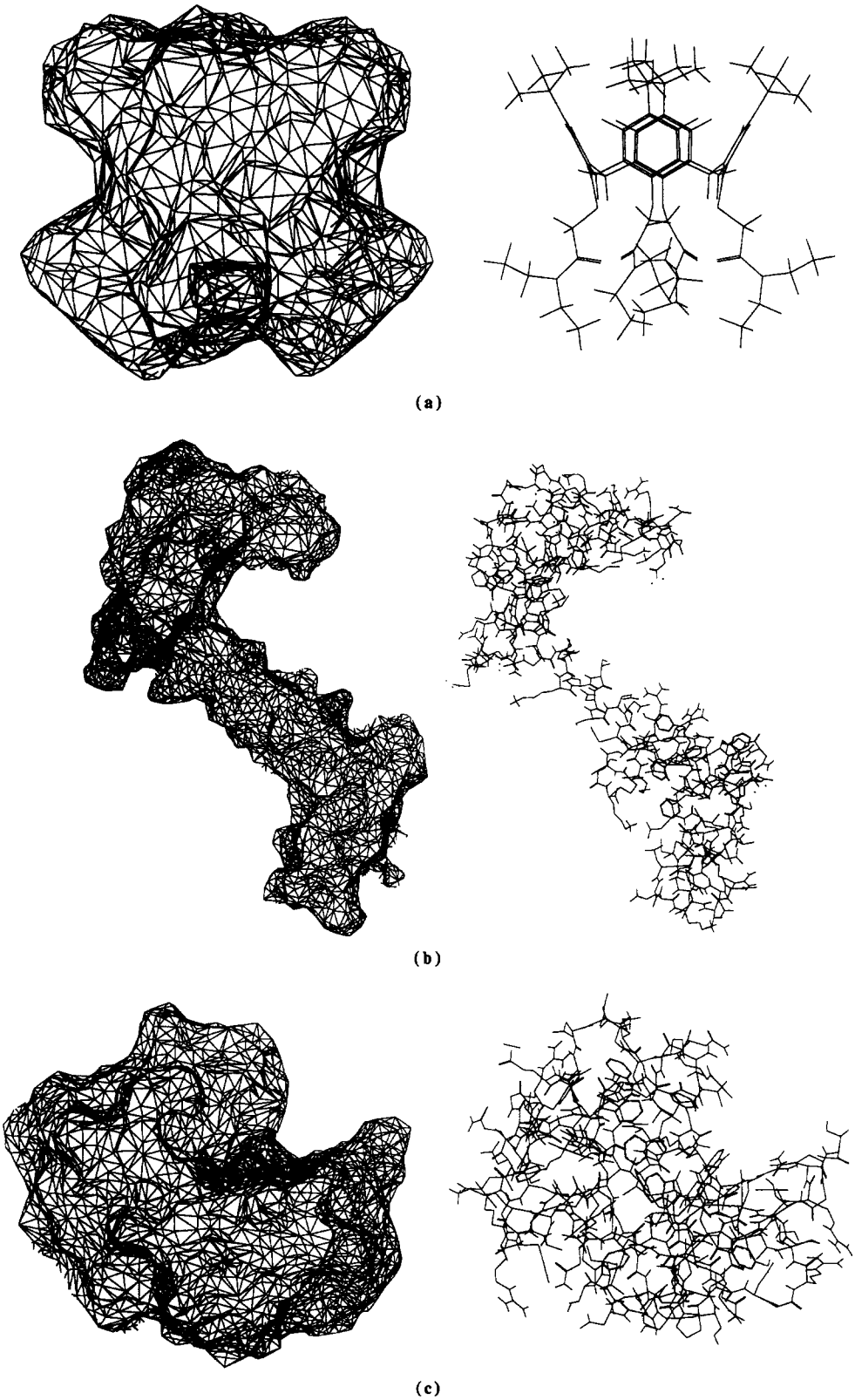


FIGURE 2. Surface triangulations and structures for the molecules considered here. Surface elements used in the mBEM calculations are curvilinear but are rendered as flat triangles in this diagram. (A z-clipping plane is introduced to remove the rear-facing portion of each mesh.) (a) Calixarene. (b) Calmodulin. (c) Lysozyme.

For the proteins, atomic radii and charges were assigned from the CHARMm[36] parameter set, as described previously.[30] Radii and charges for calixarene were previously developed by one of us.[10] All calculations with the multipole method were carried out on a Silicon Graphics R4000 workstation (with access to a large remote scratch disk for the full-matrix calculations).

Results

Table I reports the numbers of nodes and elements for each of the surface triangulations used in the calculations, along with surface area, node density, and maximum disk space required for storage of the full coefficient matrix. In what follows, the parameters for a calculation using the mBEM will be abbreviated (Δ, Γ, multipole level), where the multipole level is (m)onopole, (d)ipole, or (q)uadrupole.

Figure 3 compares the accuracy of solvation calculations performed with the mBEM as a function of Δ, Γ, and the multipole expansion level. The relative error is measured by the weighted percent difference between the solution vector [σ] found using the mBEM and the reference vector [σ_0], which represents the solution to the original system of equations with full coefficient matrix. It is seen that the mBEM is able to reproduce the "exact" result with essentially negligible error ($< 0.1\%$), given adequate choices for the grid parameters Δ and Γ and the multipole expansion level. As shown in Figure 3, for calixarene and lysozyme the mBEM provides excellent accuracy ($< 0.3\%$ error) with parameter sets $(5, 1, q)$ or $(3, 1, q)$, and an essentially exact result ($< 0.03\%$ error) for $(3, 2, q)$; furthermore, a level of accuracy that is acceptable for many applications ($< 1.0\%$ error) is realized with parameter sets $(5, 1, d)$, $(3, 1, d)$, and $(3, 2, m)$.

On the other hand, for calmodulin the highest level of accuracy ($< 0.03\%$) is only achieved with parameter set $(3, 2, q)$, which corresponds to a large region for explicit integration about each node (controlled by the grid increment $\Gamma = 2$), a relatively fine grid for the multipole expansions (reflected in the smaller choice of 3 Å for Δ), and the highest (quadrupole) level for the cube-centered expansions. Reasonable accuracy ($< 1.0\%$) for calmodulin is attained even at the monopole level, provided that $(\Gamma, \Delta) = (3, 2)$. We have not yet analyzed the specific reasons for the slower convergence for calmodulin with respect to variations in the grid parameters. However, it is clearly related to the high asymmetry of the molecule, which exaggerates the importance of long-range interactions in the electrostatic problem and thus increases the relative error associated with the multipole expansions. We will explore this behavior in more detail elsewhere. Here it is sufficient to note that despite the slower convergence, the mBEM is capable of providing a solution with negligible error, provided adequate choices for the grid parameters and expansion level.

CPU times for the mBEM calculations are compared in Figure 4. It is seen that computing time shows only a modest increase when moving from the monopole to dipole expansion level. In part, this is due to the fixed time required for explicit integration over elements that lie inside the "near" cubes associated with each node; this contribution to the total CPU time is fixed and relatively expensive. The steeper increase that occurs when moving to the quadrupole level is due mainly to the much greater time needed to compute the quadrupole coefficients and field contributions (relative to the monopole or dipole). Times range from minutes (calixarene) to as much as 5 h (calmodulin with parameter set $(3, 2, q)$; however, for the protein calculations, typical computing times are in the range of 1–2 h.

TABLE I.
Triangulation Parameters, Full-Matrix Results.

Molecule	# Atoms	# Nodes	# Elements	Surface Area (Å^2)	Node Density (/Å^2)	Full Matrix Size (MB)	CPU Time (min)	Time / Gauss-Seidel Iteration (s)
Calixar.	180	1,630	3,256	1,059.6	1.54	10.6	13.8	7.9
Lyso.	1,264	5,977	11,954	5,536.6	1.08	142.9	378.7	144.3
Calmod.	1,412	8,079	16,158	8,281.5	0.98	261.1	309.8	242.3

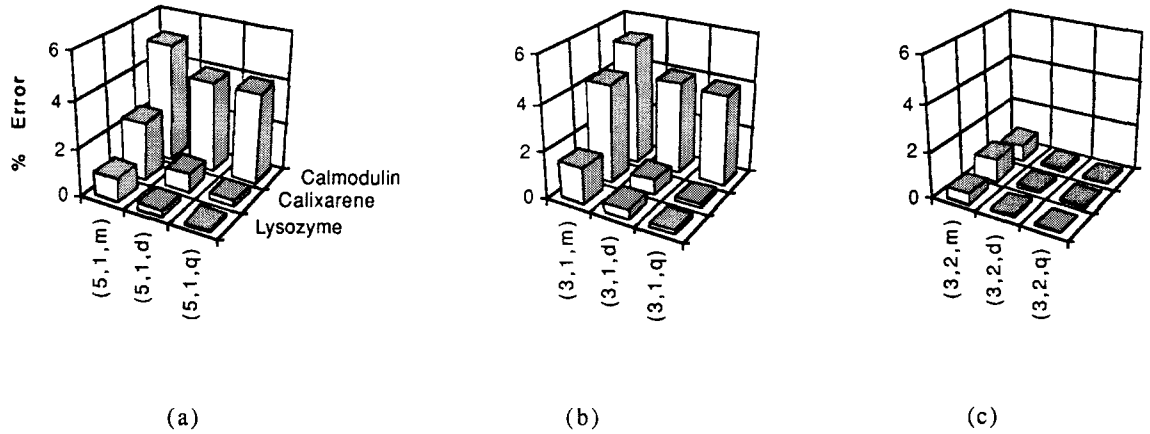


FIGURE 3. Weighted percent error of mBEM solution vectors relative to the full-matrix solutions for the three molecules considered here, for each of the multipole expansion levels. (a) $(\Delta, \Gamma) = 5, 1$. (b) $(\Delta, \Gamma) = 3, 1$. (c) $(\Delta, \Gamma) = 3, 2$.

Table I includes CPU times for the full-matrix computations. For calixarene, the mBEM method represents a speedup of 1.3–2 times over full matrix; for lysozyme the range of factors is 2.6–9 times; and for calmodulin 0.96–4.6 times. In each case, the smallest speedup (actually a slight decrease in speed for calmodulin) corresponds to a calculation at the quadrupole level, which in most cases provides a modest increase in accuracy at a large computational expense. For all the molecules considered, excellent accuracy is realized with the $(3, 2, d)$ parameter set, for which case the speedup ratios for calixarene, lysozyme, and calmodulin are 1.5, 6.2, and 2.2 times, respectively. We note that the relatively small size of calixarene leads to the involvement of a large fraction of the surface in explicit integration for all of the grid parameter sets considered—as a consequence, only a modest speedup factor can be realized for this molecule.

Figure 5a compares the CPU times required per Gauss-Seidel step in the case of lysozyme for all of the combinations of grid parameters and multipole level considered here. Figure 5b shows the total number of Gauss-Seidel steps required to achieve convergence at the 0.0001% level (weighted percent difference between successive iterations) for each of the cases presented in Figure 5a. The time per iteration at the dipole level is seen to be increased slightly relative to the monopole, while there is a dramatic increase upon moving to the

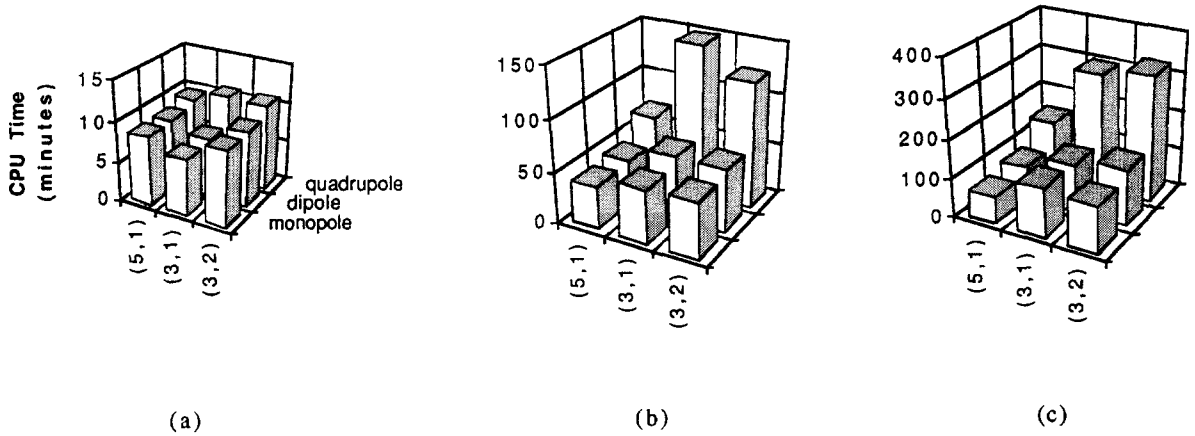
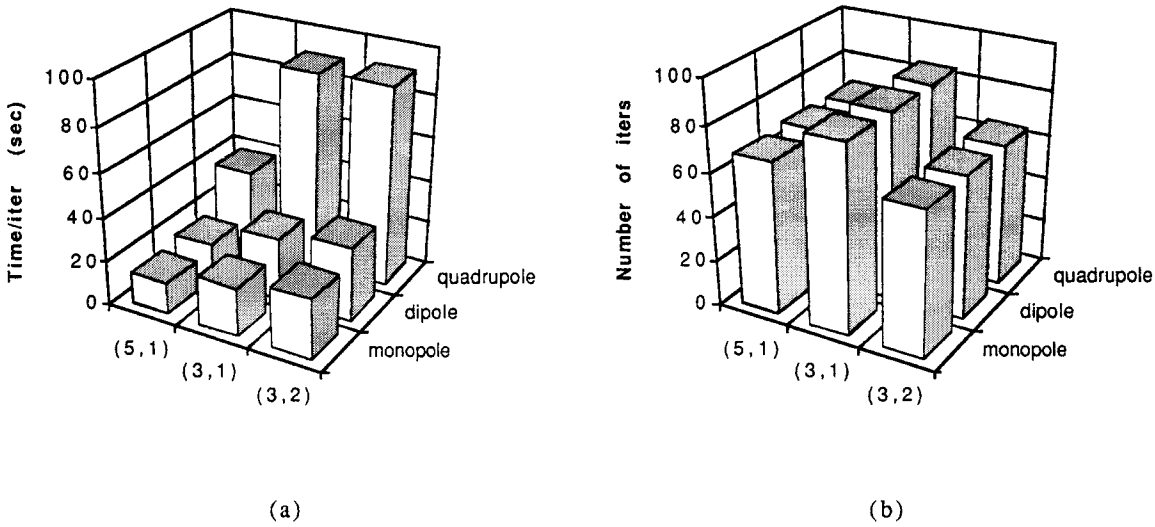


FIGURE 4. CPU times (min) for each of the mBEM calculations presented here. (a) Calixarene. (b) Lysozyme. (c) Calmodulin.



(a) (b)

FIGURE 5. (a) Time per Gauss-Seidel iteration for each of the mBEM calculations for lysozyme. Values are plotted for all combinations of grid parameters and multipole level considered. (b) Total number of Gauss-Seidel iterations required to achieve convergence for each of the parameter combinations presented in (a).

quadrupole level. This behavior, coupled with a small increase in the number of iterations for convergence observed at the dipole multipole level (Fig. 5a), gives rise to the profile of total CPU time illustrated in Figure 4. The time/iteration for the full-matrix case is 144.3 s (see Table I); the mBEM times per iteration range from 14.2 s (parameter set $(5, 1, m)$ to 90.0 s (parameter set $(3, 2, q)$).

Finally, Figure 6 reports the peak disk storage required for the three molecules and for all combinations of Γ and Δ considered here. For each molecule, the disk space requirements for parameter sets (5, 1) and (3, 2) are essentially the same, as the overall boxes enclosing the "near" elements about a given node will have approximately the same size in each case [although the (3, 2) set provides a finer grid for the multipole expansions]. With the exception of calixarene (a rather small molecule), the storage requirement for the truncated coefficient matrix in the mBEM is a fraction of that required for a full-matrix computation (see Table I).

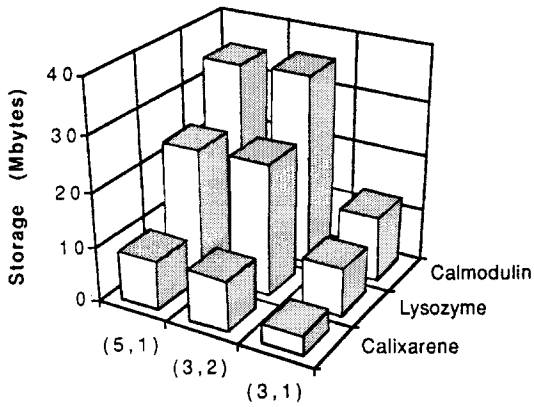


FIGURE 6. Disk storage required (in megabytes) for each of the molecules considered, as a function of the grid parameters (Δ, Γ).

Discussion

The multipole boundary element method we have presented greatly extends the applicability of methods previously developed for computing solvation and dielectric effects in large molecules. In particular, the mBEM makes it practical to handle surfaces with large numbers of vertices ($> 10{,}000$), with storage requirements that increase approximately linearly with the molecular surface area (and thus more slowly than the molecular weight). This allows macromolecules to be treated with the same level of accuracy as small molecules in continuum calculations; in contrast, FDM computations typically employ a grid of fixed size and increase numerical accuracy only in localized regions. This article has addressed only the question of the efficiency of the mBEM in reproducing the surface charge distributions that result from full-

matrix calculations. The behavior of computed solvation energies and forces with changes in the density of surface triangulations used in the mBEM (and thus the issue of numerical convergence of energies and forces) will be considered elsewhere.

An examination of Figures 3–6 shows that there is a rather wide variation in the consumption of computing resources with changes in the grid parameters Δ and Γ, and of course an overall increase with molecule size. We now present a brief qualitative analysis of this behavior, under the assumption $\Gamma = 0$ (i.e., the "near" elements for node i are just those that fall within the same grid cube as i). Let A be the total molecular surface area; ρ_a the average area density of nodes; $\rho_c(\Delta)$ (a function of the grid spacing) the average number of nodes contained in an occupied grid cube; N_0 $[= (\rho_a/\rho_c)A]$ the total number of occupied cubes; and $N_n(= \rho_a A)$ the total number of nodes. Then the storage space required is easily seen to go as

$$S \approx N_n \rho_c = \rho_a \rho_c A \tag{28}$$

Thus, the storage space increases linearly with the surface area (assuming that Δ is reasonably small compared to the largest molecular dimension) and more slowly than the molecular mass. Figure 7a illustrates the variation in the storage requirements of the mBEM as a function of molecule size (measured by the number of nodes in the surface triangulation). Although there are a limited number of data points, the graph suggests a linear relationship between these quantities.

The computing time can be expressed as the sum of two components. The first, T_L, is the time required to compute the coefficient matrix. It is expected to be proportional to the square of the number of nodes in each occupied cube times the total number of occupied cubes:

$$T_L = K_L \rho_c^2 N_0 = K_L \rho_a \rho_c A \tag{29}$$

The second component can be expressed $N_{\text{iter}} T_I$, where T_I is the time per iteration of the Gauss-Seidel/multipole algorithm, and N_{iter} is the total number of iterations required to achieve a specified level of convergence. T_I is, in turn, the sum of two parts; the time needed to multiply the coefficient matrix times the current solution vector (proportional to the number of nodes in an occupied cube times the total number of nodes), and the time needed to compute the multipole contribution to the total electric field at each node (proportional to the number of occupied cubes times the number of nodes):

$$\begin{aligned} T_I &= K_{GS} \rho_c N_n + K_M N_0 N_n \\ &= K_{GS} \rho_a \rho_c A + K_M \frac{\rho_a^2}{\rho_c} A^2 \end{aligned} \tag{30}$$

The total computing time is then given by

$$T = \rho_a \rho_c (K_L + N_{\text{iter}} K_{GS}) A + N_{\text{iter}} K_M \frac{\rho_a^2}{\rho_c} A^2 \tag{31}$$

So the total CPU time is expected to be a quadratic function of the molecular surface area. Figure 7b shows the CPU time required by the mBEM as a function of molecule size. Again, the data points, though limited in number, are consistent with the predicted behavior. The relative importance of the quadratic term in eq. (31) can be reduced by increasing the grid spacing Δ, and so

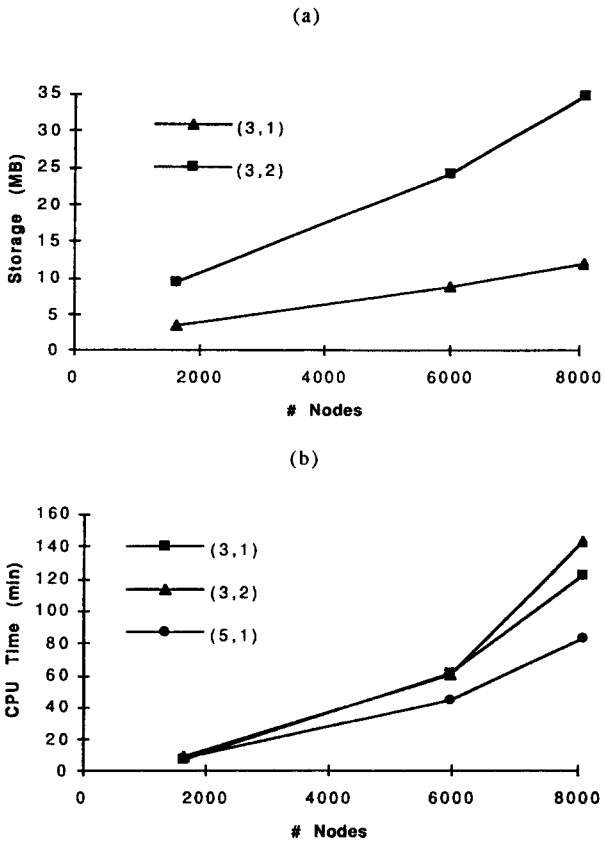


FIGURE 7. (a) Disk storage as a function of molecular surface area (measured approximately by the number of nodes). Plots are made for two combinations of grid parameters. (b) Total CPU time as a function of surface area. Plots are made for all combinations of grid parameters considered at the dipole expansion level.

reducing the number of occupied cubes. However, if Δ is made too large, then ρ_c will approach an appreciable fraction of the total number of nodes, and the first term will begin to exhibit quadratic behavior. At the same time, a large value of Δ will lead to a quadratic (rather than linear) increase in storage requirements with molecule size; evidence of this is seen in the slight nonlinearity of the curve for $\Gamma = 2$ in Figure 7. Further work will be needed to determine optimal grid parameters as a function of molecule size and node density; nonetheless, the results presented here show that computations for macromolecules can be completed in an acceptable time (1–2 h in most cases) with the mBEM, and that storage increase with molecule size will be very nearly linear.

While the times we quote for the mBEM are far in excess of those claimed for electrostatic calculations carried out using fast implementations of the FDM,[37,38] we stress that those calculations involve a grid size that is fixed and independent of the size of the molecule. In contrast, we are applying the BEM to both large and small molecules with the same level of discretization, with the goal of maintaining a consistent level of accuracy. We hope that our approach will make it practical to estimate the contribution of global hydration energetics in processes such as subunit assembly and conformational transitions.

Finally, we note that the iterative part of our algorithm employs a straightforward implementation of the Gauss-Seidel method. A technique with more rapid convergence, such as simultaneous overrelaxation,[39] might greatly reduce the overall CPU time required for the mBEM. We should also point out that we have imposed a rather severe convergence criterion here, which is certainly more stringent than required in most applications; a larger threshold would significantly reduce the CPU time consumed. We intend to explore these issues in later work.

Conclusions

We have shown that the boundary element method (BEM) can be a practical approach for the computation of solvation effects in large molecules, provided that the method is modified so that long-range interactions are approximated by a grid-based multipole expansion technique. Our new approach requires storage of only a fraction of the full coefficient matrix, while introducing only a small (or even negligible) error. By adjustment of user-selected parameters, it is possible to trade off storage and CPU requirements against numerical accuracy.

We expect that the mBEM approach we have demonstrated here will find application in those areas in which it is critical to compute solvation effects accurately for entire structures, not just localized regions. Such applications include studies of subunit assembly, large-scale conformational changes, and computation of hydration forces in large structures. We also expect that additional improvements in the algorithm will lead to significant reductions in computation time.

Acknowledgment

We thank Tripos, Inc. for providing access to computing facilities for much of this work.

References

1. P. Beroza, D. R. Fredkin, M. Y. Okamura, and G. Feher, *Proc. Natl. Acad. Sci.*, **88,** 5804 (1991).
2. D. Bashford, D. A. Case, C. Dalvit, L. Tennant, and P. E. Wright, *Biochem.*, **32,** 8045 (1993).
3. T. Takahasi, H. Nakamura, and A. Wada, *Biopolymers*, **32,** 897 (1992).
4. R. Constanciel, *Theor. Chim. Acta*, **69,** 505 (1986).
5. C. J. Cramer and D. G. Truhlar, *J. Am. Chem. Soc.*, **113,** 8305 (1991).
6. P. T. van Duijnen, A. H. Juffer, and H. P. Dijkman, *J. Mol. Struct.*, **260,** 195 (1992).
7. T. Fox, N. Rosch, and R. J. Zauhar, *J. Comput. Chem.*, **14,** 253 (1993).
8. M. E. Davis and J. A. McCammon, *Comp. Phys. Comm.*, **62,** 187 (1990).
9. R. J. Zauhar, *J. Comput. Chem.*, **12,** 575 (1991).
10. A. A. Varnek, A. S. Glebov, G. Wipff, and D. Feil, *J. Comput. Chem.*, **16,** 1 (1995).
11. M. K. Gilson and B. Honig, *Proteins*, **4,** 7 (1988).
12. B. J. Yoon and A. M. Lenhoff, *J. Phys. Chem.*, **96,** 3130 (1992).
13. J. Warwicker, *J. Mol. Biol.*, **223,** 247 (1992).
14. S. J. Shire, G. I. H. Hanania, and F. R. N. Gurd, *Biochem.*, **13,** 2967 (1974).
15. A. Karshikov, W. Bode, A. Tulinsky, and S. R. Stone, *Prot. Sci.*, **1,** 727 (1992).
16. V. Frecer and S. Miertus, *Intl. J. Quant. Chem.*, **42,** 1449 (1992).
17. J. Warwicker and H. C. Watson, *J. Mol. Biol.*, **157,** 671 (1982).
18. S. Miertus, E. Scrocco, and J. Tomasi, *Chem. Phys.*, **55,** 117 (1981).

19. R. J. Zauhar and R. S. Morgan, *J. Mol. Biol.*, **186,** 815 (1985).
20. I. Klapper, R. Hagstrom, R. Fine, K. Sharp, and B. Honig, *Proteins*, **1,** 47 (1986).
21. A. H. Juffer, E. F. F. Botta, B. A. M. van Keulen, A. van der Ploeg, and H. J. C. Berendsen, *J. Comput. Phys.*, **97,** 144 (1991).
22. M. E. Davis and J. A. McCammon, *J. Comput. Chem.*, **11,** 401 (1990).
23. M. K. Gilson, M. E. Davis, B. A. Luty, and J. A. McCammon, *J. Phys. Chem.*, **97,** 3591 (1993).
24. W. C. Still, A. Tempczyk, R. C. Hawley, and T. Hendrickson, *J. Am. Chem. Soc.*, **112,** 6127 (1990).
25. M. K. Gilson, K. A. Sharp, and B. H. Honig, *J. Comput. Chem.*, **9,** 327 (1987).
26. V. Mohan, M. E. Davis, J. A. McCammon, and B. M. Pettitt, *J. Phys. Chem.*, **96,** 6428 (1992).
27. R. J. Zauhar and R. S. Morgan, *J. Comput. Chem.*, **9,** 171 (1988).
28. R. J. Zauhar and R. S. Morgan, *J. Comput. Chem.*, **11,** 603 (1990).
29. A. M. Mathiowetz, A. Jain, N. Karasawa, and W. A. Goddard III, *Proteins*, **20,** 227 (1994).
30. R. J. Zauhar, *J. Comp-Aided Mol. Des.*, **9,** 149 (1995).
31. F. M. Richards, *Ann. Rev. Biophys. Bioeng.*, **6,** 151 (1977).
32. M. L. Connolly, *J. App. Cryst.*, **16,** 548 (1983).
33. M. L. Connolly, *J. App. Cryst.*, **18,** 499 (1985).
34. Protein Data Bank Entry 6LYZ, T. Imoto, L. N. Johnson, A. C. T. North, D. C. Phillips, and J. A. Rupley, *The Enzymes*, **7,** 665 (1972).
35. Protein Data Bank Entry 4CLN, D. A. Taylor, J. S. Sack, J. F. Maune, K. Beckingham, and F. A. Quiocho, *J. Biol. Chem.*, **266,** 21375 (1991).
36. B. R. Brooks, R. E. Bruccoleri, B. D. Olafson, D. J. States, S. Swaminathan, and M. Karplus, *J. Comput. Chem.*, **4,** 187 (1983).
37. A. Nicholls and B. Honig, *J. Comput. Chem.*, **12,** 435 (1991).
38. M. Holst and F. Saied, *J. Comput. Chem.*, **14,** 105 (1993).
39. *Numerical Recipes*, W. H. Press, B. P. Flannery, S. A. Teukolsky and W. T. Vetterling, Cambridge University Press, Cambridge, UK, 1990.